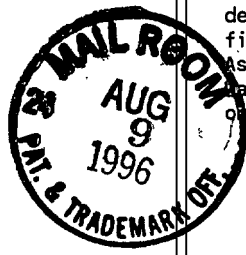


CB
2317



I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Assistant Commissioner for Patents, Washington, D.C. 20231,

PATENT

Attorney Docket No. 02307I-553

8-6-96
TOWNSEND and TOWNSEND and CREW LLP

By el Sulemt

#5/A
BH
10-9-94

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of:
MICHAEL D. DOYLE et al.
Application No.: 08/324,443
Filed: 10/17/94
For: EMBEDDED PROGRAM OBJECTS IN)
DISTRIBUTED HYPERMEDIA)
SYSTEMS)

Examiner: D. Dinh

Art Unit: 2317

AMENDMENT

RECEIVED
AUG 20 1996
GROUP 2300

Assistant Commissioner for Patents
Washington, D.C. 20231

Sir:

Responsive to the Office Action mailed May 6, 1996,
please amend the above identified application as follows:

IN THE CLAIMS:

Please cancel claim 16.

Please amend the following claims:

1. (Amended) A method for running an application program in a computer network environment, comprising:
providing at least one client workstation and one network server coupled to said network environment, wherein said network environment is a distributed hypermedia environment;
executing, at said client workstation, a browser application, that parses a distributed hypermedia document to identify text formats included in the distributed hypermedia document and for responding to predetermined text formats to initiate processes specified by the text format;

12166
10-9-94

11 utilizing said browser to display[ing], on said client
12 workstation, at least a portion of a first hypermedia document
13 received over said network from said server, wherein said first
14 hypermedia document is displayed within a first browser-
15 controlled window on said client workstation and wherein said
16 first distributed hypermedia document includes an embed text
17 format that specifies the location of an object external to the
18 first distributed hypermedia document and that specifies type
19 information utilized by said browser to identify and locate an
20 executable application external to the first distributed
21 hypermedia document.

22 invoking, with said browser application, said
23 executable application to display and process said object within
24 the first browser-controlled window while a portion of said first
25 distributed hypermedia document continues to be displayed within
26 said browser-controlled window [an embedded controllable
27 application; and

28 interactively controlling said embedded controllable
29 application from said client workstation via communications sent
30 over said distributed hypermedia environment].

1 2. (Amended) The method of claim 1, wherein said
2 executable application is a controllable application and further
3 comprising the step of:

4 interactively controlling said controllable application
5 from said client workstation via communications sent over said
6 distributed hypermedia environment [displaying is performed by
7 using a hypermedia browser application].

1 3. (Amended) The method of claim 2, wherein
2 instructions for controlling said [embedded] controllable
3 application reside on said network server, wherein said step of
4 interactively controlling said [embedded] controllable
5 application includes the following substeps:

6 issuing, from the client workstation, one or more
7 commands to the network server;

8 executing, on the network server, one or more
9 instructions in response to said commands;
10 sending information from said network server to said
11 client workstation in response to said executed instructions; and
12 processing said information at the client workstation
13 to interactively control said [embedded] controllable
14 application.

1 4. (Amended) The method of claim 2, wherein
2 instructions for controlling said [embedded] controllable
3 application reside on said client workstation.

1 5. (Amended) The method of claim 2, wherein the
2 communications to interactively control said [embedded]
3 controllable application from said client workstation continue to
4 be exchanged between the controllable application and the
5 hypermedia browser even after the controllable application
6 program has been launched.

1 6. (Amended) The method of claim 3, wherein said
2 [embedded] controllable application is a multi-dimensional
3 viewer.

1 7. (Amended) The method of claim 3, wherein said
2 [embedded] controllable application is a spreadsheet program.

1 8. (Amended) The method of claim 3, wherein said
2 [embedded] controllable application is a database program.

1 9. (Amended) The method of claim 3, wherein said
2 [embedded] controllable application is a word processor.

1 14. (Amended) The method of claim 13, wherein
2 HyperText Markup Language is used to specify said [embedded]
3 controllable application within said hypermedia document.

10/2/98 10:23:30

1 15. (Amended) A method for running an application
2 program in a computer network environment, comprising:
3 providing at least one client workstation and one
4 network server coupled to said network environment, said network
5 including a plurality of general purpose workstations, wherein
6 said network environment is a distributed hypermedia environment;
7 executing, at said client workstation, a browser
8 application, that parses a distributed hypermedia document to
9 identify text formats included in the distributed hypermedia
10 document and for responding to predetermined text formats to
11 initiate processes specified by the text format;
12 utilizing said browser to display[ing], on said client
13 workstation, at least a portion of a hypermedia document received
14 over said network from said server, wherein said hypermedia
15 document is displayed within a browser controlled window and
16 includes at least a first [embedded] text format that identifies
17 and locates a multi-dimensional data visualization application
18 external to the distributed hypermedia document;
19 invoking, with said browser application, said
20 multi-dimensional data visualization application; and
21 interactively controlling said [embedded] multi-
22 dimensional data visualization application from within said
23 browser window at said client workstation via communications sent
24 over said distributed hypermedia environment wherein data image
25 rendering is performed by said plurality of general purpose
26 workstations using distributed processing.

1 23. (Amended) The method of claim 15, wherein a
2 hypermedia browser is executing on the client workstation,
3 wherein communications to interactively control said [embedded]
4 controllable application from said client workstation continue to
5 be exchanged between the controllable application and the
6 hypermedia browser even after the controllable application
7 program has been launched.

24. (Amended) A computer controlled method for interactively controlling an [embedded] external object referred to in a document displayed in a document window on a client computer, wherein the client computer includes a processor coupled to a display device and to a user input device, wherein the processor is further coupled to a computer network, wherein the computer network is coupled to a server computer and one or more additional computers, wherein the server computer includes a local storage device containing a document, wherein the document includes a reference to an [embedded] external object, wherein an application program for manipulating the [embedded] object resides on a first additional computer, the method comprising the following steps:

transferring, over the network, at least a portion of the document from the server computer to the client computer;

parsing said document to locate a reference to the external object included in the document, with the reference identifying and locating the external object;

accepting first signals from the user input device that indicate that the [embedded] external object is to be manipulated;

issuing commands from the client computer to the first additional computer in response to the first signals;

executing, by using the first additional computer, instructions in the application program in response to the issued commands, wherein the executed instructions generate information about manipulating the [embedded] external object;

communicating, via the network, the information about manipulating the [embedded] external object from the first additional computer to the client computer;

displaying communicated information in said document window at the client computer; and

using the client computer to manipulate the [embedded] object according to the communicated information displayed.

26. (Amended) The method of claim 24, further comprising the steps of executing instructions in a second application program on a second additional computer in response to the issued commands, wherein the instructions executed by the second additional computer result in information about manipulating the [embedded] object being generated more quickly.

28. (Amended) The method of claim 26, wherein the [embedded] object is a multi-dimensional image displayable in any of a plurality of orientations.

34. (Amended) A method for displaying a three dimensional image object on a client computer, wherein the client computer includes a processor coupled to a display device, wherein the processor is further coupled to a computer network, wherein the computer network is coupled to a server computer and one or more additional computers, wherein the server computer includes a local storage device containing a hypermedia document, wherein the hypermedia document includes a reference indicating the identity and location of a three dimensional image object [embedded within] external to the hypermedia document, wherein the three dimensional image object is displayable in a plurality of orientations, the method comprising the following steps:

transferring, over the network, at least a portion of the hypermedia document from the server computer to the client computer;

displaying in a document window on the display device, by using the processor, at least a portion of the hypermedia document, wherein the displayed portion of the hypermedia document includes the three dimensional image object displayed in a first orientation;

using the client computer to issue commands over the network;

executing instruction on a first additional computer in response to the issued commands, wherein the executed instructions determine a second orientation for display of the three dimensional image object;

communicating, via the network, information about the second orientation from the first additional computer to the client computer; and

using the client computer to redisplay the three dimensional image object in the second orientation.

Please add the following claims:

44. (New) A computer program product for use in a system having at least one client workstation and one network server coupled to said network environment, wherein said network environment is a distributed hypermedia environment, the computer program product comprising:

a computer usable medium having computer readable program code physically embodied therein for causing a client workstation to invoke an external executable application referenced by a hypermedia document to display and process an external object referenced by the hypermedia document, said computer program product further comprising:

computer readable program code for causing said client workstation to execute a browser application to parse a distributed hypermedia document to identify text formats included in the distributed hypermedia document and to respond to predetermined text formats to initiate processes specified by the text format;

computer readable program code for causing said client workstation to utilize said browser to display, on said client workstation, at least a portion of a first hypermedia document received over said network from said server, wherein said first hypermedia document is displayed within a first browser-controlled window on said client workstation and wherein said first distributed hypermedia document includes an embed text format that specifies the location of an object external to the first distributed hypermedia document and that specifies type information utilized by said browser to identify and locate an executable application external to the first distributed hypermedia document;

31 computer readable program code for causing said
32 client workstation to invoke, with said browser application,
33 said executable application to display and process said
34 object within the first browser-controlled window while a
35 portion of said first distributed hypermedia document
36 continues to be displayed within said browser-controlled
37 window.

1 45. (New) The computer program product of claim 44,
2 wherein said executable application is a controllable application
3 and further comprising:

4 computer readable program code for causing said client
5 workstation to interactively control said controllable
6 application from said client workstation via communications sent
7 over said distributed hypermedia environment.

1 46. (New) The computer program product of claim 45,
2 wherein instructions for controlling said controllable
3 application reside on said network server, wherein said step of
4 interactively controlling said controllable application includes:

5 computer readable program code for causing said client
6 workstation to issue, from the client workstation, one or more
7 commands to the network server;

8 computer readable program code for causing said network
9 server to execute one or more instructions in response to said
10 commands;

11 computer readable program code for causing said network
12 sever to send information to said client workstation in response
13 to said executed instructions; and

14 computer readable program code for causing said client
15 workstation to process said information at the client workstation
16 to interactively control said controllable application.

1 47. (New) The computer program product of claim 45,
2 wherein instructions for controlling said controllable
3 application reside on said client workstation.

457 OF 4444880

computer readable program code for causing
said client work station to execute a browser

14 application, that parses a distributed hypermedia
15 document to identify text formats included in the
16 distributed hypermedia document and to respond to
17 predetermined text formats to initiate processes
18 specified by the text format;

19 computer readable program code for causing
20 said client work station to utilize said browser to
21 display at least a portion of a hypermedia document
22 received over said network from said server, wherein
23 said hypermedia document is displayed within a browser
24 controlled window and includes at least a first text
25 format that identifies and locates a multi-dimensional
26 data visualization application external to the
27 distributed hypermedia document;

28 computer readable program code for causing
29 said client work station to invoke, through said
30 browser application, said multi-dimensional data
31 visualization application; and

32 computer readable program code for causing
33 said client work station to interactively control said
34 multi-dimensional data visualization application from
35 within said browser window at said client workstation
36 via communications sent over said distributed
37 hypermedia environment wherein data image rendering is
38 performed by said plurality of general purpose
39 workstations using distributed processing.

1 55. (New) A computer program product for use in a
2 system including a client computer which includes a processor
3 coupled to a display device and to a user input device, wherein
4 the processor is further coupled to a computer network, wherein
5 the computer network is coupled to a server computer and one or
6 more additional computers, wherein the server computer includes a
7 local storage device containing a hypermedia document, wherein
8 the hypermedia document includes a reference to an external
9 object, wherein an application program for manipulating the

object resides on a first additional computer, the computer program product comprising:

a computer usable medium having computer readable program code physically embodied therein for interactively controlling an external object referred to in a hypermedia document displayed in a document window on a client computer, said computer program product further comprising:

computer readable program code for causing said network to transfer at least a portion of the hypermedia document from the server computer to the client computer;

computer readable program code for causing said client work station to parse said hypermedia document to locate a reference to the external object included in the hypermedia document, with the reference identifying and locating the external object;

computer readable program code for causing said client work station to accept first signals from the user input device that indicate that the external object is to be manipulated;

computer readable program code for causing said client work station to issue commands from the client computer to the first additional computer in response to the first signals;

computer readable program code for causing said first additional computer to execute instructions in the application program in response to the issued commands, wherein the executed instructions generate information about manipulating the external object;

computer readable program code for causing said first additional computer to communicate, via the network, the information about manipulating the external object from the first additional computer to the client computer;

computer readable program code for causing said client work station to display communicated

10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45

information in said document window at the client
computer; and
computer readable program code for causing
said client work station to manipulate the object
according to the communicated information displayed.

56. (New) A computer program product for use in a
system including a client computer which includes a processor
coupled to a display device, wherein the processor is further
coupled to a computer network, wherein the computer network is
coupled to a server computer and one or more additional
computers, wherein the server computer includes a local storage
device containing a hypermedia document, wherein the hypermedia
document includes a reference indicating the identity and
location of a three dimensional image object external to the
hypermedia document, wherein the three dimensional image object
is displayable in a plurality of orientations, the computer
program product comprising:

a computer usable medium having computer readable
program code physically embodied therein for displaying a three
dimensional image external object referred to in a document
displayed in a document window on a client computer, said
computer program product further comprising:

computer readable program code for causing
said network to transfer at least a portion of the
hypermedia document from the server computer to the
client computer;

computer readable program code for causing
said client workstation to display, in a document
window on the display device, by using the processor,
at least a portion of the hypermedia document, wherein
the displayed portion of the hypermedia document
includes the three dimensional image object displayed
in a first orientation;

computer readable program code for using the
client computer to issue commands over the network;

083444-0744
b6 b7C E-111222

31 computer readable program code for executing
32 instruction on a first additional computer in response
33 to the issued commands, wherein the executed
34 instructions determine a second orientation for display
35 of the three dimensional image object;
36 computer readable program code for
37 communicating, via the network, information about the
38 second orientation from the first additional computer
39 to the client computer; and
40 computer readable program code for using the
41 client computer to redisplay the three dimensional
42 image object in the second orientation. *wa*

REMARKS

Claims 1-43 have been examined, claims 1-9, 14, 15, 23, 24, 26, 28 and 34 are amended, claim 16 is cancelled, and claims 44-56 have been added. Accordingly, claims 1-15 and 17-56 are now pending in the application. Reexamination and reconsideration are requested.

Claims 1-4 and 15-16 are rejected under 35 U.S.C. Sec. 102(a) as being anticipated by the University of Southern California's Mercury Project (in public use from Sept. 15, 1994 to March 31, 1995 at <http://www.usc.edu/dept/raiders/>).

The present invention, as defined for example in amended claim 1, includes the steps of executing, at the client workstation, a browser application that parses a distributed hypermedia document to identify text formats included in the distributed hypermedia document and for responding to text formats to initiate processes specified by that text format. The browser is also utilized to display at least a portion of the distributed hypermedia document within a browser-controlled window.

The distributed hypermedia document includes an embed text format the specifies the location of an object external to the distributed hypermedia document and that specifies type information utilized by the browser to identify and locate an executable application external to the distributed hypermedia

464704-CH2ES0

document. The browser invokes the executable application to display and process the object within the browser window.

The Mercury Project is an interactive Web page that utilizes CGI (Common Gateway Interface) scripts and the HTML <FORM> tag to facilitate interaction between the user and the Web page.

CGI is a protocol that allows a Web server and an external program to communicate. An example is a form which utilizes the following HTML format:

```
<FORM
ACTION="identity of program to invoke on server"
METHOD="specify method of reading info"
>
<INPUT
TYPE="type"
NAME="name"
SIZE="size"
>

etc.
</FORM>
```

Information is entered by the user and passed through the Web server to the CGI server identified by the ACTION= attribute. The CGI program generates a new result Web page which is sent back to the browser and displayed in place of the original page displayed in the browser-window.

For example, at the bottom of page 2 of the cited document it is stated that after clicking on the map the arm will move and a new image will be returned.

The HTML of the page 2 of the cited Mercury Project document is appended to this amendment. The page has been decommissioned so the <FORM> tags have been removed. However, it is apparent that all images are in-line GIF and MPEG images identified by their extensions. Also, appended is a page linked to the Mercury page, the Telegarden, page that utilizes the <FORM> tag to execute control.

The <FORM> tag causes the browser to send a string of characters, entered into a form in the original HTML document, to the Web server application. The Web server invokes a CGI application identified by the ACTION= attribute of the tag and passes the string to the CGI application. In response to the character string the server application generates and sends a new HTML document to be displayed by the browser in place of the ordinal HTML document.

For example, in the Mercury Project when the arm is moved by clicking on an in-line image included in a first HTML document, displayed in a first browser-controlled window, a string is sent to the Web server with information on how far the arm is moved. The Web server invokes the CGI server which generates a new HTML document with an in-line image showing the arm in the new position and sends the new document to the browser to be displayed in place of the first HTML document. Some browsers would cache the first HTML document so that the first and new HTML documents could be viewed in different windows.

In the Telegarden page, a new HTML document including a revised image of the garden is sent by the server and displayed by the browser in place of the original document. Also, forms are very popular in search pages where a new HTML document is sent by the sever and displayed by the browser in place of the original document to show the results of the search.

The examiner states that the Mercury Project operated using a method substantially as claimed. This rejection is respectfully traversed for the following reasons.

There is no disclosure of the claimed step of utilizing a browser to display a first hypermedia document in a first window with the hypermedia document including a tag format specifying the location of an external object and an external executable application. As described above, the Mercury Project utilizes CGI where a <FORM> tag identifies a program on the server but not an external object. Additionally, the claimed step of invoking the executable object to display and process the object within the first browser-controlled window while a portion of the first hypermedia document is displayed is not disclosed.

104707 ETTA200

In the Mercury Project a new HTML document is generated at the server and displayed by the browser in a new window.

The claimed combination is fundamentally different from the Mercury Project. In the claimed combination, the external object and executable object are embedded by reference in the HTML document and the object is displayed and processed within the same window where a portion of the original document is displayed. In the Mercury Project information is passed back to the server and a new document is generated and displayed. There is no display and processing the external object within the window in which a portion of the original document is displayed.

A major difference between CGI and the claimed combination is that in CGI there is no enforced continuity between the documents. The CGI responds to form information by generating new documents each being a static document independent of the previous document which generated the form string passed to the Web server. For example, in the Mercury Project separate, independent HTML documents are generated for each position of the arm. There is no disclosure of the claimed step of invoking the executable application to display and process said object within the window while a portion of the first distributed hypermedia document is displayed in the window.

Thus, unlike CGI, the claimed executable application does not generate a static HTML document to be displayed in place of the first document but displays and processes the object in a portion of the window.

Claims 1-43 are rejected under 35 U.S.C. Sec. 103 as being unpatentable over Applicant's disclosed prior art and further in view of Hansen "Enhancing Documents with Embedded Programs: How Ness Extends Insets in the Andrew Toolkit".

The Hansen reference discloses embedding an executable script in a document. The Ness script is a sequence of attribute specifications, i.e., declarations of global variables, global functions, and **extend** blocks. An extend block associates a set of contained attributes with some named object and has the following syntax:

extend <name>

to be continued

<attributes>

end extend

where <name> must be a string constant giving the name of the associated object. (Page 25). One attribute is an event specification such a mouse click.

The operation of Ness scripts is illustrated in the "Extended Birthday Card" example at pages 30 and 31. Note that the object named is: **extend** "visible cake". Both the executable script and the object to be manipulated are within the document.

In view of the above, it is believed that the claims are not obvious over the disclosed prior art in view of Hansen. There is no disclosure in the references, singly or in combination, of displaying a hypermedia document in a first window including a text format specifying the location of an external object and identifying an external executable application or of invoking the external application to display and process the external object within the first window.

The system of Ness provides for interaction with an object embedded in a document by executing code embedded in the document. However, there is no teaching or suggestion of the claimed system of utilizing a browser to invoke an external application identified by an original document, being displayed by a browser within a first window, to display and process an external object within the first window.

In view of the above, claim 1 is deemed to be patentable over the cited references.

Claim 15 recites that the browser application invokes an external application which is a multi-dimensional visualization application. The multi-dimensional visualization application is controlled from within the browser window via communications sent over the distributed hypermedia environment.

As described in the application, a 3-D image, located external to the original document, may be displayed within the first window and processed by an executable application external to the browser.

Thus, unlike the Mercury Project, a multi-dimensional object may be displayed and manipulated within the browser window

46277 E 442230

by an application running on a remote computer. Also, Ness is distinguished because in the Ness system the application code is included in the hypermedia document. Thus, the combination claimed in claim 15 is a significant advance over systems such as disclosed in Hanson or the Mercury Project and is not suggested by any of the cited references.

Claim 24 recites accepting first signals from a user input device indicating that the external object, referenced in a hypermedia document displayed in a document window on a client computer, is to be manipulated. Commands are issued to a first additional computer in response to the first signal and the first additional computer responds to the commands by executing instructions in the application program to generate information about manipulating the external object. The information is communicated to the client computer which displays the information in the document window and is used to manipulate the object according to the information displayed.

As described above, in the Mercury Project an object in a document window can not be manipulated. The user can indicate a desired change to the object, e.g., move the arm, which is communicated to the Web server and transferred to the CGI server. The CGI server then generates a new document with an image showing the arm moved and transfers the new document back to the client which displays the new document in place of the original document. Thus, the Mercury Project does not provide for interaction with an object displayed in browser window. Additionally, as discussed in the specification, the requirement of sending a new hypermedia document to display each image change consumes bandwidth on the network.

The Ness disclosure does not teach using an application on a first additional computer to manipulate an object external to the hypermedia document within document window. In Ness both the object and application are included within the document displayed. While this feature enhances interactivity with documents, the application is executed on the computer displaying the document.

In contrast, in the combination of claim 24 the first computer could be much more powerful than the client computer to run an application performing, for example, 3-D visualization or CAD/CAM programs, and the results could be displayed in the document window of client computer in the form of a PC or network computer.

Accordingly, the subject matter of claim 24 is not disclosed or suggested by the cited references, singly or in combination.

In claim 24, the hypermedia document displayed in a document window on the client computer references an external object which is a three dimensional object to be displayed in a plurality of orientations. The external object is displayed in a first orientation within the document window. Commands are issued to a first additional computer in response to the first signal and the first additional computer responds to the commands by execute instructions in the application program to generate information about a second orientation of the external object. The information is communicated to the client computer which redisplay the external object in the second orientation within the document window.

The subject matter of claim 34 differs from the cited references for reasons similar to those discussed above with reference to claim 24. Again, the claimed system allows for a much more powerful first additional computer to perform calculations to reorient an object with the results communicated to the client computer to redisplay the object in a new orientation.

The claims added herein are of similar scope to the examined claims but are recited in the form of computer readable code, embodied on a computer readable medium, for causing a computer to effect the novel steps recited in the method claims previously examined.

In view of the foregoing, Applicants believe all claims now pending in this application are in condition for allowance. The issuance of a formal Notice of Allowance at an early date is respectfully requested.

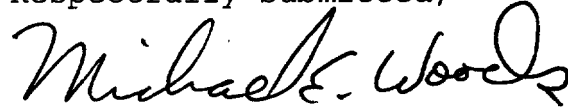
16207-444280

MICHAEL D. DOYLE et al.
Application No.: 08/324,443
Page 20

PATENT

If the Examiner believes a telephone conference would expedite prosecution of this application, please telephone the undersigned at (415) 576-0200.

Respectfully submitted,



Charles E. Krueger
Reg. No. 30,077

TOWNSEND and TOWNSEND and CREW LLP
Two Embarcadero Center, 8th Floor
San Francisco, California 94111-3834
(415) 576-0200
Fax (415) 576-0300
CEK:db
s:\023071\553\amend.01

162707-244430



TOWNSEND and TOWNSEND and CREW LLP
Two Embarcadero Center, 8th Floor
San Francisco, CA 94111-3834
(415) 578-0200

In re application of MICHAEL D. DOYLE et al.

Appln. No. 08/324,443

Filed 10/17/94

Group Art Unit 2317

For EMBEDDED PROGRAM OBJECTS IN
DISTRIBUTED HYPERMEDIA SYSTEMS

THE ASSISTANT COMMISSIONER FOR PATENTS
Washington, D.C. 20231

Sir:

Transmitted herewith is an amendment in the above-identified application.

- ☐ Enclosed is a petition to extend time to respond.
- ☒ Small entity status of this application under 37 CFR 1.9 and 1.27 has been established by a verified statement previously submitted.
- ☐ A verified statement to establish small entity status under 37 CFR 1.9 and 1.27 is enclosed.
- ☒ A document entitled *USC Mercury Project: Interface*
- If any extension of time is needed, then this response should be considered a petition therefor.
- The filing fee has been calculated as shown below:

(Col. 1)		(Col. 2)		(Col. 3)	SMALL ENTITY		OR	OTHER THAN A SMALL ENTITY	
	CLAIMS REMAINING AFTER AMENDMENT		HIGHEST NO. PREVIOUSLY PAID FOR	PRESENT EXTRA	RATE	ADDIT. FEE		RATE	ADDIT. FEE
TOTAL	56*	MINUS	43**	= 13	x11=	\$143		x22=	\$
INDEP.	8*	MINUS	4***	= 4	x39=	\$156		x78=	\$
<input type="checkbox"/> FIRST PRESENTATION OF MULTIPLE DEP. CLAIM					+125=	\$		+250=	\$
					TOTAL	\$299	OR	TOTAL	\$

* If the entry in Col. 1 is less than the entry in Col. 2, write "0" in Col. 3.

** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, write "20" in this space.

*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, write "3" in this space.
The "Highest Number Previously Paid For" (Total or Independent) is the highest number found from the equivalent box in Col. 1 of a prior amendment or the number of claims originally filed.

☐ No fee is due.

Please charge Deposit Account No. 20-1430 as follows:

- ☒ Claims fee \$ 299.00
- ☒ Any additional fees associated with this paper or during the pendency of this application.

TWO extra copies of this sheet are enclosed.

Atty. Docket No. 02307I-055300US

Date August 6, 1996

I hereby certify that this is being deposited with the United States Postal Service as first class mail in an envelope addressed to:

Assistant Commissioner for Patents
Washington, D. C. 20231.

Date:

8-6-96
J. Bullen
RECEIVED
AUG 20 1996
GROUP 2300

TOWNSEND and TOWNSEND and CREW LLP

Michael E. Woods / Reg. No.: 33,466
Attorneys for Applicant